DC 3.2 Distributed Computing Functional Requirements

Section DC 3.2.1 specifies fundamental or common requirements for distributing computing which are not specific to either of the two technologies that are being focused upon.

Section DC 3.2.2 specifies requirements that are specific to DCE.

Section DC 3.2.3 specifies requirements that are specific to CORBA.

As mentioned earlier, the distributed computing component of the COE supports the two fundamental, industry standard technologies for distributed computing, DCE and CORBA. DCE supports a *remote procedure call* (RPC) paradigm of software development, whereas CORBA supports a *distributed object management* (DOM) paradigm. There are, however, requirements that are fundamental or at a higher level than either DCE or CORBA, or which are common to both paradigms, which are specified in Section DC 3.2.1. DCE specific requirements are specified in Section 3.2.2, and CORBA specific requirements are specified in Section DC 3.2.3, below.

Some fundamental requirements for items like Remote Procedure Call or Object Request Broker functionality are not specified herein since such requirements are inherent in the DCE ad CORBA technologies that have been selected for implementation in the COE.

NOTE: in the following material, the word "implementation" is used to refer to the general set of combined capabilities used to implement the distributed computing requirements. The implementation may include both COTS and GOTS components. Additionally, the word "participant" is used to refer to entities that make use of the distributed computing implementation (in DCE terminology, these are generally referred to as principals). The word "domain" is used to generally refer to a DCE cell or a CORBA namespace, which typically correspond to a system management or security scope.

DC 3.2.1 Fundamental or Common Requirements

The requirements specified in this section of the SRS are fundamental or common to the implementation of distributed computing in the COE, and apply to the implementation of both of the DCE and CORBA technologies (including related infrastructure services) in the COE.

DC 3.2.1.1 Security

3.2.1.1.1 General Security Requirements. The implementation shall comply with, or support, those security requirements specified in the Security SRS for the COE that are applicable to the distributed computing component, including at least the following:

a. Mandatory access control [TBD]

Traceability:
Priority ???

b. Discretionary access control

Traceability:
Priority ???

c. Mutual Identification and Authentication

Traceability:
Priority ???

d. Authorization

Traceability:
Priority ???

e. Privacy

Traceability:
Priority ???

f. Integrity

Traceability:
Priority ???

g. Non-Repudiation

Traceability:
Priority ???

h. Auditing

Traceability:
Priority ???

3.2.1.1.2 Fortezza Integration. The implementation shall provide a FORTEZZA/MISSI compliant alternative encryption mechanism that is usable by all of the applicable distributed computing services.

Traceability:
Priority ???

3.2.1.1.3 Firewalls. The implementation shall provide support for use through firewalls and guards.

Traceability:
Priority ???

NOTE: Operation of the implementation through firewalls and guards is probably not a requirement that can be directly satisfied by the distributed computing component of the COE, but is more likely to be satisfied through the configuration of the entire system/network, including routers, packet filtering, intermediate hosts, etc. Even so, the implementation of distributed computing shouldn't deny service in such configurations.

DC 3.2.1.2 Dynamic Reconfigurability

3.2.1.2.1 Location Independence. The implementation shall be able to determine the location of resources by using a location-independent name to permit a client participant to bind to a resource regardless of its physical location (e.g., to support relocation of services to another host).

Traceability:
Priority ???

3.2.1.2.2 Replication. The implementation shall provide the ability to replicate its own servers (i.e., those that support the distributed computing implementation itself) and application-level services (e.g., a map server, correlation server, etc) to support fault tolerant operation and optimal performance.

Traceability:
Priority ???

3.2.1.2.3 Flexible Domain Configuration. The implementation shall have the flexibility to support a variety of deployment options, including the ability to subdivide a domain (e.g., cell or namespace) and incorporate the subdivided resources into a foreign domain and/or be hosted on a different (e.g., deployed) network.

Traceability:
Priority ???

3.2.1.2.4 Dynamic Addressing. The implementation shall support configurations where hosts [including hosts that provide distributed computing servers?] in the domain are frequently moved and must determine their network IP address at boot-up using the technique known as dynamic IP addressing.

Traceability:
Priority ???

3.2.1.2.5 Mobile Operation. [TBD]. It has yet to be determined if a requirement exists for the ability for the distributed computing implementation to support operation while moving (e.g., like cell phone operation).

Traceability:
Priority ???

DC 3.2.1.3 Synchronized Time

3.2.1.3.1 Time Service. The implementation shall provide for automatic, secure, global synchronized time.

Traceability:
Priority ???

DC 3.2.1.4 Performance and Architecture

DC 3.2.1.4.1 Scalability

NOTE: in the following requirements, rough, order-of-magnitude estimates of maximums have been provided. These are highly subjective. The range of scalability is a factor that should be considered as part of the COE product evaluation/recommendation cycle.

3.2.1.4.1.1 Intra/Inter-net/Remote Network Scalability. The implementation shall support scalable operation over intra-, internet, and remotely connected networks, proportionate to network speed and available capacity. At the low end, for remotely connected hosts, a minimum of 9.6Kb/s line speed with varying available capacity should be assumed.

Traceability:
Priority ???

3.2.1.4.1.2 Domain Scalability. The implementation shall provide scalable performance as the number of domains (e.g., cells, namespaces) increases. A maximum of domains numbering in the low thousands should be assumed.

Traceability:
Priority ???

3.2.1.4.1.3 Usage Scalability. The implementation shall provide scalable performance as the number of users/principals/servers/objects in a domain increases. Maximum site sizes may have up to 50,000 users, numbers of servers in the low hundreds, and numbers of objects in the several thousands.

Traceability:
Priority ???

NOTE: objects can exist at varying levels of granularity, and it is not difficult to imagine hundreds of millions of objects. However, it is likely that the level of granularity utilized in the COE will require more than several thousand in the COE V4-V5 timeframe.

3.2.1.4.1.4 Server Load Balancing. The implementation shall provide the ability to distribute client requests amongst replicated servers to facilitate optimal performance by exploiting any parallelism that may exist in the configuration of servers.

Traceability:
Priority ???

NOTE: DCE binding APIs provide support for the client to select from multiple available servers on either a random basis or on the basis of some other criteria that the client applies. Products like Encina extend this to provide for bindings that distribute client requests based on server load.

3.2.1.4.2 Deleted (this paragraph should be removed from the document prior to finalization).

Traceability:
Priority ???

3.2.1.4.3 Concurrency/Threading. The implementation shall provide concurrent access to services and shall support multithreading of service implementations.

Traceability:
Priority ???

DC 3.2.1.5 Fault Tolerance

3.2.1.5.1.1 Replication. The implementation shall, in the event of server failure, support: a) automatic rebinding of clients to replicated servers, where those servers are a part of the distributed computing component (e.g., DCE CDS, Security Server, etc, and b) graceful notification to the client of a server failure with the opportunity for the client to rebind to a replicated server if appropriate for the application.

Traceability:
Priority ???

3.2.1.5.1.2 Server Failure Management. The implementation shall provide the capability of monitoring the health of the servers, and be capable of automatically restarting failed servers.

Traceability:
Priority ???

3.2.1.5.1.3 Reliability. The implementation shall provide the ability to guarantee that requests for services are implemented reliably, such that the requestor can know deterministically whether the request was performed by the server.

Traceability:
Priority ???

DC 3.2.1.6 Language Support

3.2.1.6.1 Ada, C, C++. The implementation shall provide support for application software development in the following programming languages: Ada'95 (including Ada Task compatibility with threading in the distributed computing component and in the operating system), ANSI C, C++.

Traceability:
Priority ???

NOTE: We may need to specify compiler products for Ada'95 and C++ since these languages are currently either incompletely implemented or not described by formal specifications, respectively.

3.2.1.6.2 Java (Future Requirement). Support for application software development in the Java language will likely become a requirement in the future. It will probably be required that DCE or OMG IDL can be compiled to produce Java bytecode classes that can access the DCE or CORBA interface implementations using the respective distributed computing mechanisms.

Traceability:
Priority ???

3.2.1.6.2 The implementation shall provide support for an application software development language that is portable across all COE platforms.

Traceability:
Priority ???

3.2.1.6.2.1 The language shall be capable of producing executables that are platform independent, capable of running on any COE platform without recompiling or relinking.

Traceability:
Priority ???

3.2.1.6.2.2 The language shall be object oriented, supporting object oriented features such as encapsulation, inheritance, overloading, and polymorphism.

Traceability:
Priority ???

3.2.1.6.2.3 The language shall have the capability to create network connections and access objects across networks via URLs using the http protocol. This capability shall also include the ability to download machine independent bytecode executables via the http protocol.

Traceability:
Priority ???

3.2.1.6.2.4 The language shall provide multithreading capabilities.

Traceability:
Priority ???

3.2.1.6.2.5 The language shall provide the capability to generate language classes based on interfaces defined with DCE IDL and CORBA IDL. The classes generated shall be capable of accessing the DCE or CORBA interface implementations, using the respective distributed computing mechanisms.

Traceability:
Priority ???

DC 3.2.1.7 Transaction Processing

3.2.1.7.1 Atomicity. The implementation shall provide support of atomicity for ensuring that a computation consisting of one or more operations on one or more objects satisfies the requirements of atomicity (if a transaction is interrupted by a failure, any partially completed results are undone).

Traceability:
Priority ???

3.2.1.7.2 Isolation. The implementation shall provide the ability of transactions to execute concurrently, with the same result as if they were performed sequentially.

Traceability:
Priority ???

3.2.1.7.3 Durability. The implementation shall provide support for durability (if a transaction completes successfully, the results of its operations are never lost, except in the event of catastrophe).

Traceability:
Priority ???

3.2.1.7.4 Database Support. The implementation shall provide support for 3-tier applications including support for multiple databases (same or different database vendors) and multiple platforms including all of the COE platforms and database management systems.

Traceability:
Priority ???

3.2.1.7.5 Database Management Heterogeneity. The implementation shall support transactions that span across the breadth of the multiple, heterogenous database management systems that are supported by the COE (e.g. Oracle, Sybase...).

Traceability:
Priority ???

3.2.1.7.6 Process Spanning Transactions. The implementation shall provide the ability to have transactions span across multiple processes (e.g. Process A starts a transaction, Process B continues the transaction, Process A completes the transaction).

Traceability:
Priority ???

3.2.1.7.7 Database Independent API. The implementation shall provide an transaction processing API that is independent of the database management system.

Traceability:
Priority ???

3.2.1.7.8 Transaction Rollback. The implementation shall provide the ability to abort transactions and cause all involved databases to rollback to their initial state (before transaction began).

Traceability:
Priority ???

3.2.1.7.9 Nested Transactions. The implementation shall provide the ability to nest transactions.

Traceability:
Priority ???

DC 3.2.1.8 Queueing

3.2.1.8.1 Persistence. The implementation shall maintain an object in a queue until it has been de-queued, and shall provide for reliable recovery of queue contents in the event of a system restart or failure.

Traceability:
Priority ???

3.2.1.8.2 Queue Query. The implementation shall provide the capability to access (read) queued objects while they are still in the queue (i.e. you should not have to de-queue an object before you can read it).

Traceability:
Priority ???

NOTE: Current application level uses of queueing functionality require performance of at least 30 operations per second. Because performance is relative to a wide variety of factors unrelated to queueing, performance should be part of product evaluation criteria, but is probably inappropriate to define as a requirement.

3.2.1.8.3 Deleted. (This paragraph should be removed before this document is finalized).

Traceability:
Priority ???

3.2.1.8.4 Priorities. The queue service must support queuing and de-queuing of objects at at least 10 different priority levels.

Traceability:
Priority ???

3.2.1.8.5 Queue Polling. The implementation shall support: a) Synchronous Blocking, where a queue is polled and the process is blocked until something arrives in the queue; b) Synchronous Non-Blocking, where a queue is polled and control is immediately returned to the process whether there are any objects in the queue or not; and c) Asynchronous, where a queue is polled, control is immediately returned to the process, but the process is notified (at some later time) when an object arrives on the queue.

Traceability:
Priority ???

3.2.1.8.6 Queue Size. The implementation shall be capable of accomodating queue objects of at least 50K bytes.

Traceability:
Priority ???

3.2.1.8.7 Number of Queues. The implementation shall be able to create and maintain a minimum of 100 (simultaneous) queues.

Traceability:
Priority ???

3.2.1.8.8 Concurrent access. The implementation shall support: a) concurrent access to queues, b) simultaneous queueing from multiple processes, and c) multiple processes reading from the same queue.

Traceability:
Priority ???

3.2.1.8.9 Multiple Queues. The implementation shall support the ability for a process to have multiple (incoming) queues.

Traceability:
Priority ???

3.2.1.8.10 Queued Object Size. The implementation shall be capable of accommodating queue objects of a least 50K bytes.

Traceability:
Priority ???

3.2.1.8.11 Access Control. The implementation shall support the ability to define access control lists on a per queue basis.

Traceability:
Priority ???

3.2.1.8.12 Queue Locking. The implementation shall support the ability to lock queue entries to protect against concurrent write access.

Traceability:
Priority ???

3.2.1.8.13 LIFO. The implementation shall provide the ability to access the queue in last-in-first-out (LIFO) order.

Traceability:
Priority ???

NOTE: The recommended solution from Transarc does not satisfy the above requirement for LIFO access.

DC 3.2.1.9 Management

3.2.1.9.1 License Management. The implementation shall provide tools for managing any licensing mechanisms that are required by the implementation. These tools should preferably be GUI based, and should be easy to use by systems administrators. Any license management should be integrated with the othe relevant management functions.

Traceability:
Priority ???

3.2.1.9.2 Transaction Management. The implementation shall provide an administrative application to monitor transactions, including performance, failed transactions, and status of servers, and to start/stop the transaction processing monitor.

Traceability:
Priority ???

3.2.1.9.3 Reserved.

3.2.1.9.4 Queue Management. The implementation shall provide an administrative application to monitor queues, providing the following capabilities:

a. Report the number of objects in each queue.

Traceability:
Priority ???

b. Report all processes connected to each queue (process name and machine it is running on).

Traceability:
Priority ???

c. Report the status of each process (e.g. waiting, reading, writing)

Traceability:
Priority ???

d. Flush any or all queues.

Traceability:
Priority ???

e. Start and stop the queue service servers.

Traceability:
Priority ???

3.2.1.9.5 Reserved.

3.2.1.9.6 Reserved.

DC 3.2.1.10 Segmentation

The implementation shall be segmented in accordance with the version of the DII COE Integration and Run-Time Environment Specification (I&RTES) that is approved by DISA for COE 4.0.

DC 3.2.1.11 Standards

3.2.1.11.1 Standards Compliance. The implementation shall adhere with formal, industry de facto, and community standards such as the Joint Technical Architecture (JTA).

Traceability:
Priority ???

DC 3.2.1.12 Platforms

3.2.1.12.1 COE Supported Platforms. Unless otherwise stated, the full implementation (all of the capabilities described below for each paradigm) shall be supported on each of the platforms identified as DII COE supported platforms. As of this writing (COE V3.0), this list includes: Sun Solaris 2.4 and 2.5; Hewlet-Packard HP-UX 9.0.7 and 10.01; Microsoft NT 3.51 (client side software only). For COE V4.0, this list may expand to include other platforms such as Digital Unix and IBM AIX.

Traceability:
Priority ???

DC 3.2.1.13 Legacy Compatibility

3.2.1.13.1 Legacy Compatibility. The implementation shall be consistent with the requirements of legacy and migration systems that will utilize the COE.

Traceability:
Priority ???

DC 3.2.1.14 Product Quality.

3.2.1.14.1 Product Quality. The implementation shall be of a quality consistent with the best commercial practices of the industry, including continuing product improvement, bug fixes, telephone and email support, documentation, interoperability, and training.

Traceability:
Priority ???

DC 3.2.1.15 Training

3.2.1.15.1 Training. The implementation shall provid]e product training materials suitable to support the following types of training:

a. Product specific training

Traceability:
Priority ???

b. COE specific training

Traceability:
Priority ???

c. Installation training

Traceability:
Priority ???

d. System management training

Traceability:
Priority ???

e. Software development training.

Traceability:
Priority ???

DC 3.2.1.16 Documentation

3.2.1.16.1 Developer Documentation. The implementation shall include documentation to describe the proper or recommended usage of the implementation by software developers, as well as explicitly identify usage which is prohibited by the architectural tenets of the COE or which would be incompatible with other COE capabilities.

Traceability:
Priority ???

DC 3.2.2 DCE Specific Requirements

To address distributed computing for the remote procedural call based software development paradigm, the DII has adopted the Distributed Computing Environment (DCE) technology, defined by the Open Group (previously the Open Software Foundation). There are many reasons why DCE was selected,

most of which are beyond the scope of this SRS. The DII COE specific requirements for the use of DCE are specified in the next sections.

DC 3.2.2.1 DCE Version

3.2.2.1.1 Version Compliance. The implementation shall be compliant with OSF DCE V1.2.1.
Traceability:
Priority ???

DC 3.2.2.2 DCE services

3.2.2.2.1 Threads. The implementation shall use the native operating system threads services. If the operating system implementation of threads is unsupported, the DCE implementation shall provide an implementation of the POSIX pthreads services.
Traceability:
Priority ???

3.2.2.2.2 Naming. The implementation shall provide the DCE Cell Directory Service and utilize the DNS directory service for locating other cells.
Traceability:
Priority ???

NOTE: X.500 based Global Directory Services (GDS) are not subject to widespread commercial availability, due in part to the nearly ubiquitous deployment of DNS in the commercial and DoD communities. However, the COE may be required to support X.500 based GDS in the future to align with other major government initiatives such as the Defense Message System which is utilizing X.500.

3.2.2.2.3 Security. The implementation shall provide the DCE Security Service.
Traceability:
Priority ???

3.2.2.2.4 Reserved.

3.2.2.2.5 Transitive trust. The implementation shall provide transitive trust between hierarchical cells, such that principals may access services located in other cells without the need for pair-wise registration of principals between cells.
Traceability:
Priority ???

NOTE: The currently recommended DCE product from Transarc does not satisfy the transitive trust requirement. Transitive trust has not yet been implemented by the Open Group; This situation should be remedied by the COE V4.0 timeframe.

3.2.2.2.6 Reserved.

3.2.2.2.7 Reserved.

3.2.2.2.8 Time. The implementation shall provide a DCE Distributed Time Service (DTS) that is capable of interfacing with NTP for time synchronization outside of the cell.
Traceability:
Priority ???

3.2.2.2.9 Host. The implementation shall provide the DCE Host services.
Traceability:
Priority ???

DC 3.2.2.3 DCE Applications

3.2.2.3.1 Distributed File System: The implementation shall provide the DCE Distributed File System (DFS), including at least the DFS Client and DFS Exporter functionality.

Traceability:
Priority ???

3.2.2.3.2 NFS/DFS Gateway. The implementation shall provide a gateway that permits hosts to use NFS to access files in the DFS.

Traceability:
Priority ???

DC 3.2.2.4 DCE Software Development

3.2.2.4.1 Reserved.

3.2.2.4.2 Application programming interface: The implementation shall provide the API implementation of the standard DCE API and Generic Security Service API (GSS-API).

Traceability:
Priority ???

3.2.2.4.3 C++ class interface: The implementation shall provide a C++ class library interface to the DCE and GSS APIs described above for use with the C++ programming language.

Traceability:
Priority ???

NOTE: This is just a C++ interface to the DCE and GSS APIs, not a more generalized way of invoking C++ objects across a network (that is where CORBA is used).

3.2.2.4.4 DCE traffic monitor/debugger: The implementation shall provide a GUI-based tool for monitoring DCE traffic between clients and servers that can be used to assist in debugging the clients, servers, and DCE configuration.

Traceability:
Priority ???

3.2.2.4.5 Templates: The implementation shall provide example client and server software templates that demonstrate typical usage of the DCE capabilities, for each of the supported programming languages.

Traceability:
Priority ???

NOTE: This would seem to duplicate the capabilities of the ease of use APIs listed in section 3.2.2.4.8, but may be needed to support users who cannot afford to procure the ease of use libraries.

3.2.2.4.6 Reserved.

3.2.2.4.7 Ease of use APIs. The implementation shall provide an API that provides a more abstract, higher level interface to the common DCE programming idioms, such as client and server registration and initialization, obtaining a server binding, performing name/directory service lookups, use of security services (including ACL access, monitoring, and auditing), use of the GSS-API, use of RPCs, and access to identification and authentication information.

Traceability:
Priority ???

DC 3.2.2.5 Management

3.2.2.5.1 Cell Management. The implementation shall provide GUI based tools to support both local and remote (but within the cell) configuration or reconfiguration of DCE services, including support for:

a. Add/delete a host from a DCE cell.
Traceability:
Priority ???

b. Relocate a host from one cell to another.
Traceability:
Priority ???

c. Attaching a host to another network.
Traceability:
Priority ???

d. Maintenance of dynamic IP addressing capabilities.
Traceability:
Priority ???

e. Start/Stop servers.
Traceability:
Priority ???

f. Add/Delete/Modify and configure servers in a cell.
Traceability:
Priority ???

g. Maintenance of cell and lan profiles.
Traceability:
Priority ???

h. Maintaince of the endpoint map of hosts.
Traceability:
Priority ???

i. Monitor server status and be capable of starting or restarting servers upon reboot, failure, or as part of normal operating procedures.
Traceability:
Priority ???

j. Maintain the security server, including the registry, groups, access control lists, and all attributes associated with each.
Traceability:
Priority ???

k. Synchronize the security registry contents with the related operating system information, including user account and group information.
Traceability:
Priority ???

l. Maintain the time server.
Traceability:
Priority ???

m. Add/Delete/Modify filesets in the distributed file system.
Traceability:
Priority ???

n. Synchronize master and replicated servers.

Traceability:
Priority ???

o. Rebuild master servers after failure.

Traceability:
Priority ???

p. Relocate servers to different hosts.

Traceability:
Priority ???

q. Create and maintain DCE server replicas, including the security server, time server, DFS filesystem servers, and cell directory server

Traceability:
Priority ???

r. Create and maintain the hierarchical Cell Directory Service, including all of the attributes associated with the CDS.

Traceability:
Priority ???

s. Remote management of the implementation.

Traceability:
Priority ???

t. Maintain the audit functionality, including the collection, synchronization, reduction, and archival of audit logs and the start/stop of the DCE audit daemons on hosts.

Traceability:
Priority ???

u. Register cells for inter-cell authentication.

Traceability:
Priority ???

v. Backup and restore DCE server data, including at least the CDS directory and security server data.

Traceability:
Priority ???

w. Browse and search the CDS namespace.

Traceability:
Priority ???

DC 3.2.2.6 Compatibility and Migration Support

3.2.2.6.1 3-Tier Migration. The implementation shall provide tools to ease migration of legacy and 2-tier applications to a 3-tier architecture.

Traceability:
Priority ???

3.2.2.6.2 Network Protocols. The implementation shall provide the ability to add DCE's security functionality to the following network protocols, such that the protocols exhibit the security benefits of DCE and remain compatible with operating system supplied versions of these same protocols:

a. Remote Shell (rsh/rshd)

Traceability:
Priority ???

b. Remote Execution (rexec/rexecd)

Traceability:
Priority ???

c. Remote Login (rlogin,rlogind)

Traceability:
Priority ???

d. Remote Copy (rcp, rcpd)

Traceability:
Priority ???

e. Telnet

Traceability:
Priority ???

f. FTP

Traceability:
Priority ???

g. SMTP

Traceability:
Priority ???

h. SNMP

Traceability:
Priority ???

i. HTTP

Traceability:
Priority ???

3.2.2.6.3 Wrappers. The implementation shall provide example, or template, DCE wrappers that show how to encapsulate an existing non-DCE executable application such that it can be invoked via DCE, including wrapper backends that perform:

a. Command line based service invocation.

Traceability:
Priority ???

b. Other (TBD).

Traceability:
Priority ???

DC 3.2.2.7 DCE Default Configuration

The implementation shall provide a default configuration for the DCE Cell Directory Service (including namespace configuration in accordance with the DII COE DCE Implementation Plan) and DCE Security Server (including default principals and configuration to implement DII security policies).

DC 3.2.2.8 DCE Documentation

3.2.2.8.1 Concept of Operation. The implementation shall provide a DII COE DCE Concept of Operations (CONOPS) document describing the overall definition, role, operation and maintenance of DCE cells in the DII. This is a higher level document than the DII DCE Adminstration Guide.

Traceability:
Priority ???

3.2.2.8.2 Administration Guide. The implementation shall provide a DII COE DCE Administration Guide that describes procedures and tools for the details of daily administration of a DCE cell and how to interconnect DCE cells within and across, organizational boundaries.

Traceability:
Priority ???

DC 3.2.3 CORBA Specific Requirements

To address distributed computing needs for the object-oriented software development paradigm, the DII has adopted the Common Object Request Broker (CORBA) technology, defined by the Object Management Group (OMG). There are many reasons why CORBA was selected, most of which are beyond the scope of this SRS. The DII COE specific requirements for the use of CORBA are specified in the next sections.

DC 3.2.3.1 Background

This section is included for information purposes only at this time, until CORBA requirements are transition are better understood. In the future, this section should be removed from this document.

Programs that are planning, designing, or using CORBA include:

a. New Attack Submarine NSSN program. This program has specified CORBA for use in interfacing its subsystems over the C3I System network. The Prime contractor is Lockheed Martin Federal Systems Division. Lockheed Martin proposed using IONA. It will pobably be the only ORB product used in the system. No work on object class definitions has been done yet. NSSN has many applications that will be based upon processors other than workstations such as: HP 743I VME board running HPRT and PowerPCs running VxWorks.
b. Theater Battle Management Core Systems. The prime contractor for TBMCS (LORAL) has chosen IONA ORBIX as the ORB for design/implementation. TBMCS will integrate CTAPS, CIS, and WCCS under a single architecture.
c. DARPA Distributed Air Operations Center (DAOC) Advanced Technology Demonstration (ATD). Logicon has selected IONA ORBIX as the commercial ORB.
d. DARPA/ISO - Joint Task Force ATD program - provides collaborative tools for the CJTF and staff. Linked with theater CINCS and deployed forces. The architectural contractor, Teknowledge Federal Systems, has been supporting a two-ORB policy, using IONA ORBIX as the commercial ORB and Corbus, a GOTS ORB developed by BBN. The system is currently moving to a second commercial ORB that has not been selected.
e. JFACC Program - just getting underway, will provide a collaborative capability for the JFACC and staff that enables a continuous planning cycle for employment of air assets. The JFACC program will use the JTF ATD architecture as a starting point (described above).

COMMENT: The following requirements were received from Navy, but I donÕt think that schedule is within the scope of an SRS. Are the wrapper development efforts dependent upon CORBA being in the COE kernel? Or is it that the Navy would like to wrap the referenced products using the CORBA products recommended by the DCWG and so the Navy wants a product decision by the specified timeframe?

The time frame in which systems require CORBA technology vary; The Navy desires that some of its applications, including NIPS, TDBM, and ATWCS be wrapped with CORBA wrappers by November 1996. The Air Force's TBMCS program is currently using CORBA in design/development and will deploy some operational CORBA based capabilities beginning in 4QCY97.

DC 3.2.3.2 CORBA Version

The implementation shall be compliant with version 2.0 of the CORBA specifications, as specified by the Object Management Group.

Note: There is not currently a validation and compliance testing suite, so compliance right now is not something that can easily be verified.

DC 3.2.3.3 CORBA Interfaces

3.2.3.3.1 CORBA. The implementation shall provide implementations of the following adopted CORBA interfaces:

a. ORB core

Traceability:
Priority ???

b. IIOP

Traceability:
Priority ???

c. Implementation Repository

Traceability:
Priority ???

d. Interface Repository

Traceability:
Priority ???

e. IDL compiler

Traceability:
Priority ???

f. Static Invocation Interface

Traceability:
Priority ???

g. Dynamic Invocation Interface

Traceability:
Priority ???

h. Dynamic Skeleton Interface.

Traceability:
Priority ???

3.2.3.3.2 CORBAservices. The implementation shall provide the following CORBAservices as defined by the OMG:

a. Naming

Traceability:
Priority ???

b. Event Management

Traceability:
Priority ???

c. Transaction

Traceability:
Priority ???

d. Lifecycle

Traceability:
Priority ???

e. Security

Traceability:
Priority ???

f. Query

Traceability:
Priority ???

g. Time

Traceability:
Priority ???

Note: Some of the CORBAservices specified above have not yet been implemented by vendors, although they have been adopted by the OMG. Those that are specified for COE V4.0 are expected to be available within the needed time frame.

3.2.3.3.3 Future CORBA Services. In the future, the implementation shall provide the following additional CORBAservices:

a. Concurrency

Traceability:
Priority ???

b. Relationship

Traceability:
Priority ???

c. Licensing

Traceability:
Priority ???

d. Persistence

Traceability:
Priority ???

e. Trader

Traceability:
Priority ???

f. Properties

Traceability:
Priority ???

g. Externalization.

Traceability:
Priority ???

DC 3.2.3.4 CORBAfacilities

3.2.3.4.1 CORBAfacilities. The implementation shall provide the following CORBAfacilities as specified by the OMG: a) Compound Document Presentation and Data Interchange. This facility is based on the Opendoc specifications developed by IBM, Apple, CIL, et al.

Traceability:
Priority ???

DC 3.2.3.5 CORBA Applications

3.2.3.5.1 Interface Repository Browser: The implementation shall provide a GUI-based capability for browsing the interfaces that are contained in the interface repositories of local and remote systems, as permitted by security policy.

Traceability:
Priority ???

DC 3.2.3.6 CORBA Software Development

3.2.3.6.1 Inter-ORB traffic monitor/debugger: The implementation shall provide a GUI-based tool for monitoring CORBA traffic between clients and servers that can be used to assist in debugging the clients, servers, and CORBA configuration.

Traceability:
Priority ???

3.2.3.6.2 Templates: The implementation shall provide example client and server software templates that demonstrate typical usage of the common CORBA interfaces, for each of the supported programming languages.

Traceability:
Priority ???

DC 3.2.3.7 Management

3.2.3.7.1 Implementation repository management: The implementation shall provide a GUI-based tool for managing the contents and configuration of local and remote implementation repositories.

Traceability:
Priority ???

3.2.3.7.2 Interface repository management: The implementation shall provide a GUI-based tool for managing the contents and configuration of local and remote interface repositories.

Traceability:
Priority ???

3.2.3.7.3 Namespace management: The implementation shall provide a GUI-based tool for managing the contents and configuration of the CORBA namespace.

Traceability:
Priority ???

3.2.3.7.4 Security management: The implementation shall provide a GUI-based tool for managing the security configuration of the CORBA implementation.

Traceability:
Priority ???

DC 3.2.3.8 Compatibility and Migration Support

3.2.3.8.1 DCE Compatability. The implementation shall be compatible with the COE implementation of the DCE.

Traceability:
Priority ???

NOTE: To the extent possible, the CORBA and DCE implementations should leverage off of each other's strengths, and CORBA capabilities should re-use or be layered upon the DCE implementation such that duplication is minimized and greater consistency is obtained.

3.2.3.8.2 Application Service Wrapping. The implementation shall provide the capability to access DCE enabled application services.

Traceability:
Priority ???

Note: The above might take the form of a CORBA/DCE generic bridge, or might involve the wrapping of DCE application services with CORBA wrappers. The practical ability to accomplish this will probably have to be determined on a case-by-case basis, depending on the DCE services that the application services uses, such as DCE pipes and pointers.

3.2.3.8.3 Microsoft Distributed Common Object Model (DCOM). The implementation shall the capability for OLE objects to request services from CORBA objects and vice-versa, using CORBA adopted technology.

Traceability:
Priority ???

DC 3.2.4 Requirements Submitted by the Army

DC 3.2.4.1 Distributed Computing shall provide the capability to retrieve each temporal event which has expired and return it to the requesting applicatoni program.

Traceability: ARMY, 20 July 1996
Priority ???

DC 3.2.4.2 The FAAD Force Operations workstation requires the ability to participate in two distinct cells simultaneiously. One cell interacts with the ATCCS system and the other interacts with the FAAD C2 Engagement Operations workstation.

Traceability: ARMY, 20 July 1996
Priority ???